HW1 graded, solutions on website
Short lecture today
Project presentations next week, discuss after lecture
Outline

1. Recap of Last Two Lectures
2. A Reduction to Approximation Algorithm Design for Welfare
3. Conclusion
Outline

1. Recap of Last Two Lectures
2. A Reduction to Approximation Algorithm Design for Welfare
3. Conclusion
We considered Single-parameter problems in a Bayesian setting.

**Bayesian Assumption**

We assume each player’s value is drawn independently from some distribution $F_i$.

We sought BIC mechanisms.

**Examples**

- Single-item Auction
- $k$-item Auction
- Position Auctions
- Matching
- Knapsack
- Single-minded CA
Revenue-optimal Mechanisms

First, we considered the revenue objective,

**Lemma (Myerson’s Virtual Surplus Lemma)**

Fix a single-parameter problem, and let $M = (A, p)$ be a BIC mechanism where a player bidding zero pays nothing in expectation. The expected revenue of $M$ is equal to the expected ironed virtual welfare served by $A$.

**Theorem**

For any single-parameter problem, where player’s private parameters are drawn independently, the revenue-maximizing auction is that which maximizes ironed virtual welfare.
First, we considered the revenue objective,

**Lemma (Myerson’s Virtual Surplus Lemma)**

Fix a single-parameter problem, and let $M = (A, p)$ be a BIC mechanism where a player bidding zero pays nothing in expectation. The expected revenue of $M$ is equal to the expected ironed virtual welfare served by $A$.

**Theorem**

For any single-parameter problem, where player’s private parameters are drawn independently, the revenue-maximizing auction is that which maximizes ironed virtual welfare.

**Implication**

Enables optimal auction implementation when the welfare-maximization problem is tractable, such as in the single-item auction, $k$-item auction, matching, etc.
We have identified the revenue optimal mechanism for arbitrary single-parameter problems, however this is not helpful for problems where [virtual] welfare maximization is NP-hard

- e.g. Single-minded CA, Knapsack

**Corollary**

*If a single parameter problem admits a polynomial time DSIC $\alpha$-approximation (worst case) mechanism for welfare, then it also admits a polynomial-time DSIC $\alpha$-approximation (average case) mechanism for revenue.*

- e.g. we saw $\sqrt{m}$ for Single-minded CA, 2 for Knapsack
BIC Approximate Mechanisms for Single-Parameter Problems

- For DSIC, when approximation was necessary, we have designed IC mechanisms carefully catered to the problem.
- In the Bayesian setting, requiring only BIC, we showed a generic reduction.
  - Used the ironing idea used for revenue maximization.
For DSIC, when approximation was necessary, we have designed IC mechanisms carefully catered to the problem.

In the Bayesian setting, requiring only BIC, we showed a generic reduction.

- Used the ironing idea used for revenue maximization

**Theorem (Hartline, Lucier 10)**

For any single-parameter problem where player values are drawn independently from a product distribution $F$ supported on $[0, 1]^n$, any allocation algorithm $A$, any parameter $\epsilon$, there is a BIC algorithm $\overline{A}_\epsilon$ that preserves the average case welfare of $A$ up to an additive $\epsilon$, and moreover can be implemented in time polynomial in $n$ and $\frac{1}{\epsilon}$. 

Recap of Last Two Lectures
A (weak) generalization of the HL10 result to multi-parameter problems: a reduction from BIC approximate welfare maximization to non-IC welfare-maximization approximation algorithms.

A brief overview of current/future trends in bayesian AMD.

Course recap
Outline

1 Recap of Last Two Lectures

2 A Reduction to Approximation Algorithm Design for Welfare

3 Conclusion
## Setup and Assumptions

### Bayesian Mechanism Design Problem in Quasi-linear Settings

Public (common knowledge) inputs describes

- **Set** $\Omega$ of *allocations*.
- **Typespace** $T_i$ for each player $i$.  
  - $T = T_1 \times T_2 \times \ldots \times T_n$
- **Valuation map** $v_i : T_i \times \Omega \rightarrow \mathbb{R}$ for each player $i$.
  - For type $t \in T_i$, denote by $v_i^t : \Omega \rightarrow \mathbb{R}$
- **Distribution** $\mathcal{D}$ on $T$

### Additional Assumptions

- Distribution $\mathcal{D} = F_1 \times \ldots \times F_n$, where $F_i$ is distribution of player $i$'s type.
- Each type-space $T_i$ is finite and given explicitly. Same for the associated prior $F_i$.
- The objective is Social welfare
- Bounded valuations $v^t_i(\omega) \in [0, 1]$
Bayesian Mechanism Design Problem in Quasi-linear Settings

Public (common knowledge) inputs describes

- Set $\Omega$ of allocations.
- Typespace $T_i$ for each player $i$.
  - $T = T_1 \times T_2 \times \ldots \times T_n$
- Valuation map $v_i : T_i \times \Omega \rightarrow \mathbb{R}$ for each player $i$.
  - For type $t \in T_i$, denote by $v^t_i : \Omega \rightarrow \mathbb{R}$
- Distribution $\mathcal{D}$ on $T$

Additional Assumptions

- $\mathcal{D} = F_1 \times \ldots \times F_n$, where $F_i$ is distribution of player $i$’s type
- Each type-space $T_i$ is finite and given explicitly. Same for the associated prior $F_i$.
- The objective is Social welfare
- Bounded valuations $v^t_i(\omega) \in [0, 1]$
Example: Generalized Assignment

- \( n \) self-interested agents (the players), \( m \) machines.
- \( s_i(j) \) is the size of player \( i \)'s task on machine \( j \). (public)
- \( C_j \) is machine \( j \)'s capacity. (public)
- \( v_i(j) \) is player \( i \)'s value for his task going on machine \( j \). (private)

**Goal**

Partial assignment of jobs to machines, respecting machine budgets, and maximizing total value of agents (welfare).

\[ T_i \] listed explicitly, each \( t \in T_i \) gives \( v_i^t : j \rightarrow \mathbb{R} \)
Example: Combinatorial Allocation

- \( n \) players, \( m \) items.
- Private valuation \( v_i : \text{set of items} \rightarrow \mathbb{R} \).
  - \( v_i(S) \) is player \( i \)'s value for bundle \( S \).

**Goal**

Partition items into sets \( S_1, S_2, \ldots, S_n \) to maximize welfare:

\[
v_1(S_1) + v_2(S_2) + \ldots v_n(S_n)
\]

- \( T_i \) listed explicitly, each \( t \in T_i \) gives \( v^t_i : 2^{[m]} \rightarrow \mathbb{R} \), either written explicitly as code, logical formulae, or an oracle.
A simplified version of a result of Bei/Huang ’11 and Hartline/Kleinberg/Malekian ’11.

**Theorem**

For any multi-parameter problem where player values are drawn independently from a product distribution $F$ supported on $[0, 1]^n$, any allocation algorithm $A$, any parameter $\epsilon$, there is an $\epsilon$-BIC algorithm $\overline{A}_\epsilon$ that preserves the average case welfare of $A$ up to an additive $\epsilon$, and moreover can be implemented in time polynomial in $n, \frac{1}{\epsilon}$, and total number of player types.
A simplified version of a result of Bei/Huang ’11 and Hartline/Kleinberg/Malekian ’11.

**Theorem**

For any multi-parameter problem where player values are drawn independently from a product distribution $F$ supported on $[0, 1]^n$, any allocation algorithm $A$, any parameter $\epsilon$, there is an $\epsilon$-BIC algorithm $\overline{A}_\epsilon$ that preserves the average case welfare of $A$ up to an additive $\epsilon$, and moreover can be implemented in time polynomial in $n$, $\frac{1}{\epsilon}$, and total number of player types.

The $\epsilon$ loss is due to random sampling technicalities which we will ignore...
Recall: The Matching Property

For each player $i$, define a bipartite graph $G_i$ with types $T_i$ on either side, and weights

$$w(t_i, t'_i) = \mathbb{E}_t \left[ v^t_i \left( A(t'_i, t_{-i}) \right) \right],$$

namely the expected value of a player of type $t_i$ for “pretending” to be of type $t'_i$.

Matching Property (Bayesian Setting, Finite typespaces.)

An allocation algorithm $A$ is said to satisfy the matching property if, for every player $i$, the identity matching $\{ (t_i, t_i) : t_i \in T_i \}$ is a maximum-weight bipartite matching in $G_i$. 

Fact (from HW2)

An allocation algorithm $A$ is implementable in Bayes-Nash equilibrium if and only if it satisfies the matching property. Truth-telling payments can be calculated as r.h.s dual variables in maximum bipartite matching problem (equivalently, VCG interpretation)
Recall: The Matching Property

For each player $i$, define a bipartite graph $G_i$ with types $T_i$ on either side, and weights

$$w(t_i, t'_i) = \mathbb{E}_{t_{-i}} [v_{t_i}^i (A(t'_i, t_{-i}))],$$

namely the expected value of a player of type $t_i$ for “pretending” to be of type $t'_i$.

### Matching Property (Bayesian Setting, Finite typespaces.)

An allocation algorithm $A$ is said to satisfy the matching property if, for every player $i$, the identity matching $\{(t_i, t_i) : t_i \in T_i\}$ is a maximum-weight bipartite matching in $G_i$.

### Fact (from HW2)

An allocation algorithm $A$ is implementable in Bayes-Nash equilibrium if and only if it satisfies the matching property.

Truth-telling payments can be calculated as r.h.s dual variables in maximum bipartite matching problem (equivalently, VCG interpretation)
We now perform a multi-parameter analogue of ironing.

**Remapping**

Fix a player $i$. Construct $\overline{A}$ which satisfies the matching property for $i$ as follows:

- Compute* maximum weight matching in $G_i$. Let $\overline{t}_i$ denote the r.h.s type matched to $t_i$, which we refer to as $t_i$’s “surrogate” type.
- Let $\overline{A}(t) = A(\overline{t}_i, t\_{\neg i})$
We now perform a multi-parameter analogue of ironing

**Remapping**

Fix a player $i$. Construct $\overline{A}$ which satisfies the matching property for $i$ as follows:

- Compute* maximum weight matching in $G_i$. Let $\overline{t}_i$ denote the r.h.s type matched to $t_i$, which we refer to as $t_i$’s “surrogate” type.
- Let $\overline{A}(t) = \overline{A}(\overline{t}_i, t_{-i})$

**Easy Fact**

$\overline{A}$ satisfies the matching property for the chosen player $i$.

Computing the dual (equivalently, VCG) prices for the matching gives truth-telling prices for player $i$. 

---

*Note: The asterisk (*) indicates a computational step that might be omitted or approximated depending on the context. This is not a standard notation and is included here for illustrative purposes.*
Wrinkle

We showed how to remap a single player’s allocation rule to restore incentive compatibility for that player, without decreasing his expected utility. Need to do all players simultaneously...

But mapping player $i$’s type $t_i \sim F_i$ to $\tilde{t}_i$ changes the weights for other player $j$’s bipartite graph! This is because $\tilde{t}_i$ is not necessarily distributed as $F_i$. 
Wrinkle

We showed how to remap a single player’s allocation rule to restore incentive compatibility for that player, without decreasing his expected utility. Need to do all players simultaneously...

But mapping player $i$’s type $t_i \sim F_i$ to $\tilde{t}_i$ changes the weights for other player $j$’s bipartite graph! This is because $\tilde{t}_i$ is not necessarily distributed as $F_i$.

Question

How can we remap all players’ types simultaneously, restoring the matching property, yet preserving the distribution of each player’s type?
We need...

For each player $i$ a (possibly random) mapping $M_i : t_i \rightarrow \bar{t}_i$ such that,

- Distribution Preservation: For $t_i \sim F_i$, we are guaranteed $\bar{t}_i \sim F_i$.
- $\overline{A}(t) = A(\bar{t}_i, t_{-i})$ satisfies the matching property for $i$
- $\mathbb{E}[v_{\bar{t}_i}^i(A(t))] \leq \mathbb{E}[v_{\bar{t}_i}^i(\overline{A}(t))]$
Attempt 2: Preserve the Distribution

We need . . .

For each player $i$ a (possibly random) mapping $M_i : t_i \rightarrow \bar{t}_i$ such that,

- Distribution Preservation: For $t_i \sim F_i$, we are guaranteed $\bar{t}_i \sim F_i$.
- $\bar{A}(t) = A(\bar{t}_i, t_{-i})$ satisfies the matching property for $i$
- $E[v^t_i(A(t))] \leq E[v^t_i(\bar{A}(t))]$

Remapping with Duplication

1. Construct a bipartite graph with a multiset of types $T_i$ on each side
   - Number of copies of $t_i$ on l.h.s proportional to $f_i(t_i)$
   - Number of copies of $s_i$ on r.h.s proportional to $f_i(s_i)$
   - Weight $w(t_i, s_i)$ is expected utility of player with type $t_i$ for pretending to be $s_i$

2. Compute* maximum weight matching.

3. Let $M_i(t_i)$ be a type $\bar{t}_i$ matched to one of the copies of $t_i$ chosen randomly.
We need...

For each player $i$ a (possibly random) mapping $M_i : t_i \rightarrow \bar{t}_i$ such that,

- Distribution Preservation: For $t_i \sim F_i$, we are guaranteed $\bar{t}_i \sim F_i$.
- $\overline{A}(t) = A(\bar{t}_i, t_{-i})$ satisfies the matching property for $i$
- $E[v_{ti}^i(A(t))] \leq E[v_{ti}^i(\overline{A}(t))]$

Equivalently: Remapping Probability Mass

1. Construct a bipartite graph with types $T_i$ on each side
   - Demand of $t_i$ on l.h.s is $f_i(t_i)$
   - Supply of $s_i$ on r.h.s is $f_i(s_i)$
   - Weight $w(t_i, s_i)$ is expected utility of player with type $t_i$ for pretending to be $s_i$

2. Compute* maximum weight flow, subject to demand and supply.

3. Let $M_i(t_i)$ be a type $\bar{t}_i$ chosen according to the flows as probabilities.
Proof: Matching Property

- Fix a player $i$, suffices to show the existence of a truth-telling payment rule for $i$.
- Intuition behind approach came from restoring matching property, but a simpler proof follows from VCG interpretation of remapping procedure.
Proof: Matching Property

- Fix a player $i$, suffices to show the existence of a truth-telling payment rule for $i$.
- Intuition behind approach came from restoring matching property, but a simpler proof follows from VCG interpretation of remapping procedure.
- A player of type $t_i$ faces an auction for “probability events”, each associated with a surrogate bid $s_i$ of value $w(t_i, s_i)$.
Proof: Matching Property

- Fix a player $i$, suffices to show the existence of a truth-telling payment rule for $i$.
- Intuition behind approach came from restoring matching property, but a simpler proof follows from VCG interpretation of remapping procedure.
- A player of type $t_i$ faces an auction for “probability events”, each associated with a surrogate bid $s_i$ of value $w(t_i, s_i)$.
- Other players in the auction: fake “replicas” of player $i$, with types given by the l.h.s types.
Proof: Matching Property

- Fix a player $i$, suffices to show the existence of a truth-telling payment rule for $i$.
- Intuition behind approach came from restoring matching property, but a simpler proof follows from VCG interpretation of remapping procedure.
- A player of type $t_i$ faces an auction for “probability events”, each associated with a surrogate bid $s_i$ of value $w(t_i, s_i)$.
- Other players in the auction: fake “replicas” of player $i$, with types given by the l.h.s types.
- The auction is constrained to allocating each event to at most one of the replicas.
Proof: Matching Property

- Fix a player $i$, suffices to show the existence of a truth-telling payment rule for $i$.
- Intuition behind approach came from restoring matching property, but a simpler proof follows from VCG interpretation of remapping procedure.
- A player of type $t_i$ faces an auction for “probability events”, each associated with a surrogate bid $s_i$ of value $w(t_i, s_i)$.
- Other players in the auction: fake “replicas” of player $i$, with types given by the l.h.s types.
- The auction is constrained to allocating each event to at most one of the replicas.
- The assignment of events to replicas is welfare maximizing, and therefore admits VCG truth-telling payments.
Proof: Matching Property

- Fix a player $i$, suffices to show the existence of a truth-telling payment rule for $i$.
- Intuition behind approach came from restoring matching property, but a simpler proof follows from VCG interpretation of remapping procedure.
- A player of type $t_i$ faces an auction for “probability events”, each associated with a surrogate bid $s_i$ of value $w(t_i, s_i)$.
- Other players in the auction: fake “replicas” of player $i$, with types given by the l.h.s types.
- The auction is constrained to allocating each event to at most one of the replicas.
- The assignment of events to replicas is welfare maximizing, and therefore admits VCG truth-telling payments.

Lemma

Applying the remapping procedure to a player $i$ results in an allocation rule that satisfies the matching property for player $i$. 
Proof: Distribution Preservation

Demand and supply constraints are such that remapping preserves the probability of each type.

Lemma

Let $\bar{t}_i = M_i(t_i)$, for $t_i \sim F_i$. It is the case that $\bar{t}_i \sim F_i$. 
The remapping procedure weakly increases welfare

**Lemma**

\[ E[v^t_i(A(t))] \leq E[v^t_i(\overline{A}(t))]. \]

This follows from the fact that the remapping computes a maximum welfare remapping of types to surrogate types, as compared to original identity mapping.
The three lemmas together imply the main theorem, after accounting for $\epsilon$ error due to sampling the weights of the edges.
The three lemmas together imply the main theorem, after accounting for $\varepsilon$ error due to sampling the weights of the edges.

**Theorem**

For any multi-parameter problem where player values are drawn independently from a product distribution $F$ supported on $[0, 1]^n$, any allocation algorithm $A$, any parameter $\varepsilon$, there is an $\varepsilon$-BIC algorithm $\overline{A}_\varepsilon$ that preserves the average case welfare of $A$ up to an additive $\varepsilon$, and moreover can be implemented in time polynomial in $n$, $\frac{1}{\varepsilon}$, and total number of player types.
Outline

1. Recap of Last Two Lectures
2. A Reduction to Approximation Algorithm Design for Welfare
3. Conclusion
In single-parameter settings, we saw that we have a mature theory
- A general reduction of BIC revenue maximization to BIC welfare maximization, approximation preserving.
- A general reduction of BIC welfare maximization to algorithm design, approximation preserving.
In single-parameter settings, we saw that we have a mature theory
- A general reduction of BIC revenue maximization to BIC welfare maximization, approximation preserving.
- A general reduction of BIC welfare maximization to algorithm design, approximation preserving.

In Multi-parameter, the picture is still in flux
- We saw a reduction from BIC welfare maximization to algorithm design, approximation preserving, only when type space is small
  - explicitly given, or constant parameters, etc
- Revenue-optimal mechanisms, and their computational complexity, remain poorly understood
  - Even in very simple settings, such as matching with i.i.d values,
  - Recent work tries to make progress on these questions.
Course Wrapup

1. Game theory and mechanism design basics
   - Games of complete and incomplete information, equilibrium concepts such as Nash equilibria, dominant strategy equilibria, Bayes-Nash equilibria
   - The mechanism design problem, the revelation principle, incentive compatibility
Prior-free Mechanism Design

- Single-parameter: monotonicity characterization, application to approximation mechanism design for combinatorial auctions, knapsack, and scheduling
- Multi-parameter problems: VCG, characterization of IC, MIR/MIDR as a paradigm for approximation mechanism design, techniques such as Lavi/Swamy LP technique and Rounding anticipation, and application to assignment problems and combinatorial auctions
Bayesian Mechanism Design

- Single-parameter: Myerson’s characterization of optimality, reduction from IC revenue maximization to IC welfare maximization, reduction from IC welfare maximization to non-IC welfare maximization.
- Multi-parameter: A conditional reduction from IC welfare maximization to non-IC welfare maximization, approximation preserving.
Next week: Project Presentations!!